Counterfactual Regret Minimization

Aydin Karsidag
Department of Electrical & Computer Engineering
McGill University
Montréal, CA H3A 0B9
skylar.karsidag@mail.mcgill.ca

Mohamad Ali Jarkas
Department of Electrical & Computer Engineering
McGill University
Montréal, CA H3A 0B9
mohamad.jarkas@mail.mcgill.ca

1 Introduction

Extensive games are a subset of games in the study of game theory which are particularly interesting because of the sequential nature of play. While there exist classic games like Rock Paper Scissors, prisoner's dilemma, and auctions, which require the players to play simultaneously, many of the popular games today which have survived centuries of history incorporate sequential turns from the players. As a result, the complexity of solving games like chess, backgammon, poker, and go, are generally extremely computationally intensive compared to their one-shot counterparts.

1.1 Motivation

As a sequential game's complexity grows, the associated game tree grows rapidly. Finding equilibrium strategies from these large game trees proves to be a difficult task. Additionally, games with incomplete information add another layer of complexity for any solution concept aiming to compute Nash equilibrium strategies.

This paper reviews foundational work done by Zinkevich et al. in [1] that introduces Counterfactual Regret Minimization (CFR), a technique that enables convergence to a Nash equilibrium through self-play. The motivation behind CFR is to provide an efficient method for solving large incomplete information extensive games. Zinkevich highlights that state of the art solution techniques at the time used linear programming with a game representation which scales linearly to the number of game states [1]. Thus, additional steps were needed to reduce the game size to a tractable number of game states. The authors mention game state abstraction and subgame partitioning methods which reduce the number of game states [1]. However, at the time of publishing, iterative techniques were shown to be capable of finding solutions to games with 10^{10} game states, a significant improvement compared to linear program based solution concepts, yet far from the 10^{18} game states in heads-up limit Texas Hold'em poker [1].

1.2 Result summary

Zinkevich et al. introduce the notion of CFR and prove that minimizing CFR minimizes overall regret [1]. This is substantive because while it is not proven in the paper, previous work has demonstrated that minimizing overall regret will lead to an approximate Nash equilibrium strategy [1]. They additionally present an algorithm which can minimize CFR generally and apply it in the context of heads-up limit Texas Hold'em [1]. The algorithm is then used to compute equilibria for various levels of abstraction up to 10^{12} game states [1]. The equilibria strategies are then played against two other strong poker programs from the

2006 AAAI Computer Poker Competition to assess their strength [1]. They conclude that their equilibrium strategies outperform the other poker programs [1].

2 Description of the model

Zinkevich et al. present a model which describes finite extensive games with imperfect information. For the purpose of this paper we will focus on the salient features of the model. These features include:

- Possible **histories** of actions denoted H. The set H describes all possible sequences of actions in the game tree including those which do not terminate at a leaf node. Z is used to specifically denote the action sequences which terminate at a leaf node. $A(h)$ is used to specify the actions available after a history $h \in H$.
- All information sets belonging to player i denoted $\mathcal{I}_i$ called the **information partition**. Each **information set** $I_i \in \mathcal{I}_i$ has the property that $A(h) = A(h')$ when h and h' share a partition.

Additionally, the work assumes two player zero-sum games with perfect recall. The model is used to build the concept of CFR which is then proven to be an upper bound for overall regret. Therefore it is important to discuss the concept of regret in games and how it is used to approximate equilibria.

2.1 Regret minimization

2.1.1 Background Information and Context

Counterfactual Regret Minimization (CFR) builds upon the concept of **regret minimization**. In the context of simple normal-form games, regret minimization is a solution concept where each player repeatedly plays the game T times following their strategy profile σ_i^t on each round t. On each round t, the player selects an action $a \in A$ according to σ_i^t to play. They then update their strategy profile σ_i^{t+1} based on the cumulative regret of not having played an alternative strategy σ_i^*. This process continues for T rounds wherein the average strategy profile $\bar{\sigma}_i^T$ approximates an equilibrium. This process is said to be regret minimizing if the average overall regret R_i^T goes to zero as T approaches infinity [1].

The **instantaneous regret** for not playing an alternative strategy σ_i^* at round t, given that the opponent played σ_{-i}^t, is:

$$R_t = u_i(\sigma_i^*, \sigma_{-i}^t) - u_i(\sigma_i^t, \sigma_{-i}^t) \tag{1}$$

This quantifies the utility difference between an alternative strategy σ_i^* and the actual strategy σ_i^t used by player i, assuming the opponent's strategy at round t is known, else we may substitute the observed actions of the opponent.

The **cumulative regret** is then defined for all actions $a \in A$ as:

$$R_{cum}^T(a) = \sum_{t=1}^{T} u_i(a, \sigma_{-i}^t) - u_i(a_i^t, \sigma_{-i}^t) \tag{2}$$

$$R_{cum}^{T,+} = max(R_{cum}^T, 0) \tag{3}$$

The cumulative regret is a running total of how much player i has built up regret towards a particular action. If a particular action has larger cumulative regret compared to other actions, the player has incentive to

incorporate that action more frequently into their strategy profile for the next round of play. It is convenient to also define the positive cumulative regret for the regret matching policy which is discussed in 2.1.2

Furthermore, the **average overall regret** over T rounds is defined as [1]:

$$R_i^T = \frac{1}{T} \max_{\sigma_i^* \in \Sigma_i} \sum_{t=1}^{T} \left[u_i(\sigma_i^*, \sigma_{-i}^t) - u_i(\sigma_i^t, \sigma_{-i}^t) \right] \tag{4}$$

This expression captures the maximum possible gain the player could have achieved on average over rounds, had they consistently played the single best fixed strategy in hindsight, rather than their actual sequence of strategies.

Finally, we define the **average strategy profile** which is currently trivial but will be amended in the context of CFR.

$$\bar{\sigma}_i^T = \frac{1}{T} \sum^{T} \sigma_i^t \tag{5}$$

It is worth highlighting that regret minimization does not solve for an exact Nash equilibrium but instead approximates an equilibrium up to a tolerance of ϵ. It is mentioned in [1] that a previous theorem specifies that:

Theorem *In a zero-sum game at time T , if both player's average overall regret is less than ϵ, then $\bar{\sigma}^T$ is a 2ϵ equilibrium.*

Thus, the quality of an equilibrium solution found from regret minimization is dependent on how much the average overall regret can be minimized. An ideal regret minimization algorithm would not only converge to zero but do so in few iterations.

2.1.2 Rock Paper Scissors example

Rock Paper Scissors is a very simple one-shot game discussed in our class. We will apply regret matching to this game to demonstrate the concept. Regret matching refers to the policy in which the player will update their strategy profile proportional to the positive cumulative regret [2]. Specifically, the strategy profile for round $t + 1$ will be equal to the normalized positive cumulative regrets in round t [2].

$$\sigma_i^{t+1} = \frac{R_{cum}^{t,+}}{|R_{cum}^{t,+}|} \tag{6}$$

	R	P	S
R	0	-1	1
P	1	0	-1
S	-1	1	0

Table 1: Normal form game of Rock Paper Scissors with payoffs for row player

Table 1 shows the payoff matrix for Rock Paper Scissors with the row player utilities. Let's explore an example where we track the row player's regrets and strategy profile starting from round 1. Table 2 outlines an example where the row player has arbitrarily chosen to begin with a pure strategy of rock and R_{cum}^t is initialized to zero (arbitrary actions have been chosen for the column player). It is typical to assign a uniform

strategy profile when the cumulative regret is non-positive [2] but for the sake of the example we have not done this.

Iteration	P_{row} strategy	P_{row} action	P_{col} action	P_{row} R_t	P_{row} R^t_{cum}
1	$(1, 0, 0)$	R	P	$(0, 1, 2)$	$(0, 1, 2)$
2	$(0, \frac{1}{3}, \frac{2}{3})$	S	S	$(1, -1, 0)$	$(1, 0, 2)$
3	$(\frac{1}{3}, 0, \frac{2}{3})$	S	P	$(-2, -1, 0)$	$(-1, -1, 2)$
4	$(0, 0, 1)$	S	R	$(1, 2, 0)$	$(0, 1, 2)$
5	$(0, \frac{1}{3}, \frac{2}{3})$	P	R	$(-1, 0, -2)$	$(-1, 1, 0)$
6	$(0, 1, 0)$	-	-	-	-

Table 2: Regret matching policy example for Rock Paper Scissors tracking the row player. Strategy and regret actions are always ordered in (Rock, Paper, Scissor).

The row player begins by selecting an action to play from their current strategy. For the first iteration this is necessarily to play rock. We then learn that the column player has played paper, thus we compute the regret of not having played paper and scissors which are 1 and 2 respectively (the regret of not having played rock is 0 since we did play rock). The resulting instantaneous regret is $(0, 1, 2)$. The instantaneous regret is then added to the cumulative regret which was initialized to 0. Finally, the row player updates their strategy profile using (6) for the next interation. Notably, iteration 4 computes a pure strategy of scissors because the only positive entry in the cumulative regret is for scissors.

We notice that if we compute the average strategy profile for the row player we get $\bar{\sigma}^6_{row} = (\frac{2}{9}, \frac{5}{18}, \frac{1}{2})$. We can imagine that if this process continued, we might eventually converge to the equilibrium $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$.

2.2 Counterfactual regret

As mentioned, counterfactual regret builds upon the concepts of regret minimization and generalizes it to extensive games with incomplete information. Therefore, some of the equations discussed in 2.1 must be generalized to accommodate sequential play and information sets.

First, Zinkevich et al. [1] introduce the concept of **reach probabilities**. A reach probability $\pi(h)$ formally defines the likelihood of reaching a history $h \in H$. These probabilities will be used to find the counterfactual utilities at a given information set and thus the immediate counterfactual regrets at every information set in the game tree. They are defined as follows:

- $\pi^\sigma(h)$ is the probability of reaching history h if all players follow a strategy σ.
- $\pi^\sigma(h, h')$ is the probability of reaching history h' from h under strategy σ.
- $\pi^\sigma_{-i}(h)$ is the probability of reaching history h when all players except i, including chance, play according to strategy σ. We do not consider the strategy of player i, which is equivalent to saying that player i intentionally plays a modified strategy to reach h [2]. This is known as the **counterfactual reach probability** since player i computes a probability which is counter to its own factual strategy.
- $\pi^\sigma(I) = \sum_{h \in I} \pi^\sigma(h)$ is the sum of reach probabilities for all histories in the information set I. $\pi^\sigma_{-i}(I)$ is defined similarly.

The authors then define the **counterfactual utility** $u_i(\sigma, I)$ as [1]:

$$u_i(\sigma, I) = \frac{\sum_{h \in I,\, h' \in Z} \pi^{\sigma}_{-i}(h)\, \pi^{\sigma}(h, h')\, u_i(h')}{\pi^{\sigma}_{-i}(I)} \tag{7}$$

The intuition is that we are interested in defining the expected utility for each information set I which considers the counterfactual probability of reaching I and subsequently the payoffs $u_i(h')$ which are reachable from I. Since $h' \in Z$, these histories are exclusively the payoffs at the leaf nodes of the game tree. The denominator ensures that the counterfactual utility considers the posterior probability of being in a history $h \in I$.

Finally, the **immediate counterfactual regret** is defined [1]:

$$R^T_{i,imm}(I) = \frac{1}{T} \max_{a \in A(I)} \sum_{t=1}^{T} \pi^{\sigma^t}_{-i}(I) \left[u_i(\sigma^t|_{I \to a}, I) - u_i(\sigma^t, I) \right] \tag{8}$$

$$R^{T,+}_{i,imm}(I) = max(R^T_{i,imm}(I), 0) \tag{9}$$

For all $a \in A(I)$, $\sigma|_{I \to a}$ is the strategy profile identical to σ except that player i always chooses action a when in information set I.

It is apparent how similar immediate counterfactual regret is to average overall regret. Here, the immediate counterfactual regret measures how much better player i could have performed, on average, by always selecting a specific action a at information set I, compared to following the original strategy σ^t. Notably, the regrets are weighted by the counterfactual reach probability of encountering I, ensuring that we compare the utility difference *if the player had tried to play actions to reach* I [1]. The positive counterfactual regret $R^{T,+}_{i,imm}(I)$ is defined similar to positive cumulative regret in 2.1.

Zinkevich et al. then prove the following [1]:

$$R^T_i \leq \sum_{I \in \mathcal{I}_i} R^{T,+}_{i,imm}(I) \tag{10}$$

We outline the proof in 3.1. The result is significant because it demonstrates that minimizing the immediate counterfactual regret for all information sets of player i will also minimize the average overall regret. As discussed in 2.1, this process can produce an equilibrium strategy of arbitrary tolerance to a Nash equilibrium.

The authors propose the use of Blackwell's algorithm for approachability [1] which has been discussed in 2.1 under the pretext of regret matching. For specific's on how Blackwell algorithm is used and how it guarantees that the regret is bounded, see [1]. We define the average counterfactual regret for each action as [1]:

$$R^T_i(I, a) = \frac{1}{T} \sum_{t=1}^{T} \pi^{\sigma^t}_{-i}(I) \left[u_i(\sigma^t|_{I \to a}, I) - u_i(\sigma^t, I) \right] \tag{11}$$

Based on these regrets, the strategy for the next iteration is defined, where $R^{T,+}_i(I, a) = \max(R^T_i(I, a), 0)$ [1]:

$$\sigma^{T+1}_i(I)(a) = \begin{cases} \dfrac{R^{T,+}_i(I, a)}{\sum_{a \in A(I)} R^{T,+}_i(I, a)} & \text{if } \sum_{a \in A(I)} R^{T,+}_i(I, a) > 0 \\ \dfrac{1}{|A(I)|} & \text{otherwise} \end{cases} \tag{12}$$

We notice that this is very similar to the regret matching policy discussed in 2.1, where the new strategy at I is the average counterfactual regret normalized across all actions $A(I)$. The final strategy approximating equilibrium would be the average strategy profile which is amended from 2.1 to include the reach probabilities.

$$\bar{\sigma}_i^t(I)(a) = \frac{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I)\sigma^t(I)(a)}{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I)} \tag{13}$$

We now move on to illustrate these equations using an example of Kuhn poker.

2.2.1 Kuhn Poker

Kuhn poker is a very simple poker form used to model a zero sum two-player imperfect information game. We are going to be using it to introduce the concept of counterfactual regret discussed in [1]. Figure 1 shows the whole game tree where the payoffs are for player I and the information sets are indicated with dashed lines. The game is limited to 3 cards, king, queen, and jack, where king beats both queen and jack while queen beats jack (K>Q>J). First, both players wager a unit of money, then nature plays and gives each player a single card keeping the third card hidden. Player I then decides to either bet an additional unit of money or pass. If they bet, player II must decide to either call the wager (which would result in both players showing their card to see who wins) or fold and forfeit the wagered money. If on the other hand player I passes, then player II has the opportunity to bet or pass. In the event that both players pass, then the cards are revealed and the winner takes the wager.

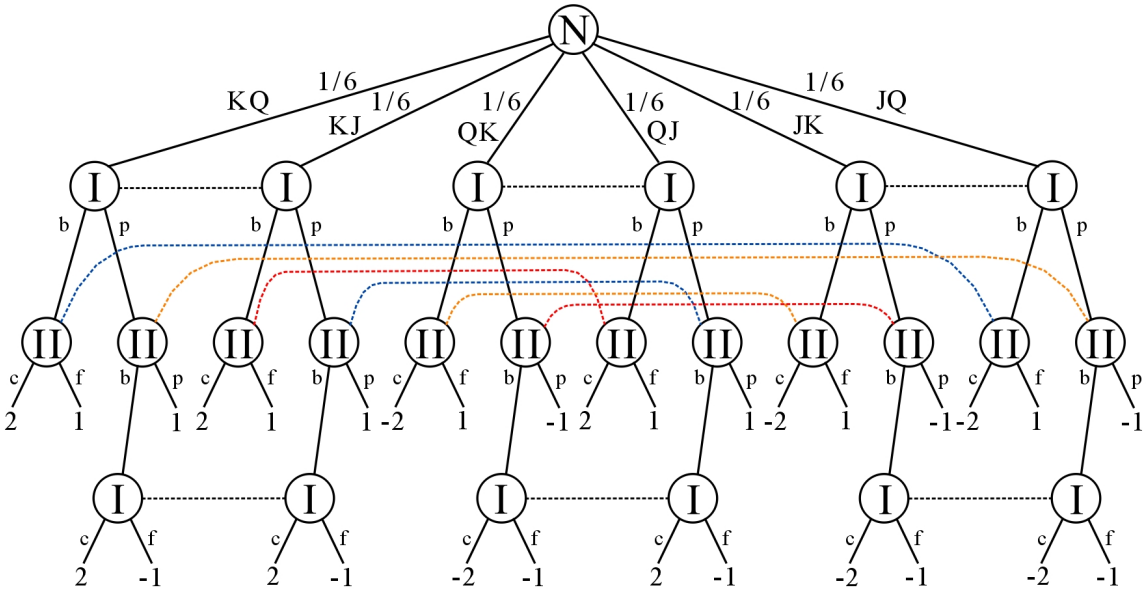

Figure 1: The game tree for 2-player Kuhn poker. Player I receives the first card and player II the second. Information sets are indicated with dashed lines.

For this example we will assume that the strategy σ^t is known and we will compute the next strategy profile $\sigma^{t+1}(I)$. Let us focus on the information set where player I receives a queen and has observed the actions pass, then bet ($I_1 = Qpb$), and decides to call with $\frac{2}{3}$ probability and fold with $\frac{1}{3}$. Further assume that player

I always first passes if they receive a queen and player II will always bet with a king when passed to and will bet with a probability of $\frac{1}{3}$ with a jack when passed to.

Begin by computing,

$$u_1(\sigma^t, Qpb) = \frac{1/6 \cdot 2/3 \cdot (-2) + 1/6 \cdot 1/3 \cdot (-1) + 1/18 \cdot 2/3 \cdot 2 + 1/18 \cdot 1/3 \cdot (-1)}{1/6 + 1/18}$$

$$u_1(\sigma^t|_{Qpb \to c}, Qpb) = \frac{1/6 \cdot 1 \cdot (-2) + 1/18 \cdot 1 \cdot 2}{1/6 + 1/18}$$

$$u_1(\sigma^t|_{Qpb \to f}, Qpb) = \frac{1/6 \cdot 1 \cdot (-1) + 1/18 \cdot 1 \cdot (-1)}{1/6 + 1/18}$$

We find that $u_1(\sigma^t, Qpb) = u_1(\sigma^t|_{Qpb \to c}, Qpb) = u_1(\sigma^t|_{Qpb \to f}, Qpb) = -1$. It is now possible to compute the counterfactual regret (for simplicity assume $T = 1$).

$$R_1^1(Qpb, c) = R_1^1(Qpb, f) = (\cdot)[-1 - (-1)]$$

In this case we find the counterfactual regret of not having played both check and fold to be zero. This means we do not regret having played our strategy at all for the first iteration. Subsequently, we would use (12) to easily compute $\sigma^2(Qpb)$, however, this is trivial once the counterfactual regret has been found. To note, the values chosen were from a Nash equilibrium solution which is why we were able to find that player I has zero counterfactual regret in only one iteration.

3 Description and proof of the result

As seen in class, iterative algorithms such as fictitious play can be employed to certain families of games to converge to equilibrium strategies. Zinkevich et al. define CFR and prove that minimizing CFR will also minimize overall regret and can thus produce an approximate equilibrium solution [1]. In this sense, CFR minimization is one such algorithm focusing on solving zero sum, perfect recall, extensive games. The strength of an iterative algorithm is based on its time complexity and convergence bounds. The authors then continue to prove that minimizing CFR through Blackwell's algorithm for approachability (regret matching) has bounds similar to Gordon's Lagrangian Hedging algorithms but does not incur a slow quadratic program projection [1]. Instead, it decomposes overall regret in a manner which allows for independent regret minimization at each information set [1]. The algorithm is then used to train an abstracted game of heads-up limit Texas Hold'em poker with 1646×10^9 game states, around 100 times larger than previous works [1]. The validity of these results were demonstrated by testing their approximate equilibria against poker programs from the 2006 AAAI Computer Poker Competition [1]. These results are covered in 4.

3.1 Proof outline

The paper provides proofs for CFR bounding the average overall regret in (10) and that the overall regret is bounded [1]. The first of these proof begins by defining the **full counterfactual regret** [1]:

$$R_{i,full}^T(I) = \frac{1}{T} \max_{\sigma' \in \Sigma_i} \sum_{t=1}^{T} \pi_{-i}^{\sigma^t}(I)(u_i(\sigma^t|_{D(I) \to \sigma'}, I) - u_i(\sigma^t, I)) \tag{14}$$

The authors then define $D(I)$ as the information sets reachable from I [1]. They also define successor operations which either return the probability of reaching an information set I' from I or the set of all information sets immediately succeeding I given that the player has played an action $a \in A(I)$ [1]. Manipulating the information set successor and reach probability operators, the relationship between full and immediate counterfactual regret is highlighted through a lemma [1]:

$$R^T_{i,full}(I) \leq R^T_{i,imm}(I) + \sum_{I' \in Succ_i(I)} R^{T,+}_{i,full}(I') \tag{15}$$

It is important to note that during this derivation the game is assumed to be perfect recall. The derivation then continues and shows that the following lemma can be proven recursively using the previous lemma [1]:

$$R^{T,+}_{i,full}(I) \leq \sum_{I' \in D(I)} R^{T,+}_{i,imm}(I') \tag{16}$$

They begin by establishing a base case where there is an information set without any successors. In such a case then the first lemma proves the second. Zinkevich et al. then derive that by using perfect recall, one can add information sets to the set of successors in a disjoint manner which proves the lemma by proof by induction [1].

The second proof establishing the convergence bound is less involved and simply argues that the regret minimization policy used in (12) is an implementation of regret matching from Blackwell's approachability theorem. Blackwell's approachability theorem states that when regret matching of that form is implemented, the bound on the regret is defined [1]:

$$\max_{a \in A} R^t(a) \leq \frac{|u|\sqrt{|A|}}{\sqrt{T}} \tag{17}$$

The authors apply the above bound in the context of CFR and conclude [1]:

$$R^T_i \leq \frac{\Delta_{u,i}|\mathcal{I}_i|\sqrt{|A_i|}}{\sqrt{T}} \tag{18}$$

Thus, the bound on the overall regret is a function of 4 variables: the range of payoffs to player i, $\Delta_{u,i}$, the size of the information partition, the maximum size of the action set possible for player i, $|A_i|$, and the iterations T. It is undesirable that the bound increases with the size of the game tree, however, if left to run for sufficient iterations, the algorithm can guarantee an arbitrarily small bound. As illustrated in the Kuhn Poker example in 2.2.1, this framework correctly evaluates regret over possible counterfactual decisions at information sets and aligns with theoretical expectations of zero-epsilon regret at equilibrium.

4 Experimental results

The theory of counterfactual regret minimization is particularly relevant to games like poker. The author uses the heads-up limit Texas Hold'em variant to experimentally demonstrate their CFR minimization algorithm. This is a two-player zero-sum game with four rounds of card dealing and betting, resulting in nearly 10^{18} game states [1]. To make the simulation computationally feasible, abstraction techniques are applied. The most critical abstraction involves merging information sets—grouping strategically similar card sequences into buckets.

This is done using a metric called hand strength squared, which accounts not only for the probability of winning but also for the variance in outcomes—favoring hands that give players clearer information before showdown. The abstraction process partitions card sequences into buckets at each round based on this metric, recursively refining them round-by-round [1]. For example, using 10 buckets (dubbed CFR-10) results in a reduced 1.65×10^{12} game states and 5.73×10^{7} information sets, which is significantly less than the 9.17×10^{17} game states and 3.19×10^{14} information sets in the full game [1]. However, this abstraction is still two orders of magnitude larger than any previously solved game [1].

To evaluate performance, the authors use the unit of millibets per hand (mb/h), where a millibet is one-thousandth of a small bet. This unit measures the exploitability of a strategy—lower mb/h indicates a strategy closer to a Nash equilibrium.

Abs	Size ($\times 10^9$)	Iterations ($\times 10^6$)	Time (h)	Exp (mb/h)
5	6.45	100	33	3.4
6	27.7	200	75	3.1
8	276	750	261	2.7
10	1646	2000	326†	2.2

†: parallel implementation with 4 CPUs

Figure 2: Number of game states, number of iterations, computation time, and exploitability of the resulting strategy for different sized abstractions [1]

Figure 2 illustrates the size, iteration count, convergence time, and exploitability associated with different abstraction levels. As the number of buckets increases, the strategy requires more time to converge but results in lower exploitability and thus a strategy that is closer to a Nash equilibrium.

	PsOpti4	S2298	CFR5	CFR6	CFR8	CFR10	Average
PsOpti4	0	-28	-36	-40	-52	-55	-35
S2298	28	0	-17	-24	-30	-36	-13
CFR5	36	17	0	-5	-13	-20	2
CFR6	40	24	5	0	-9	-14	7
CFR8	52	30	13	9	0	-6	16
CFR10	55	36	20	14	6	0	22
Max	55	36	20	14	6	0	

Figure 3: Winnings in mb/h for the row player in Texas Hold'em [1].

Figure 3 shows the performance of the different abstracted versions of the CFR bot (CFR5 through CFR-10) against other bots from the 2006 AAAI Computer Poker Competition. Notably, even the lowest-resolution abstraction, CFR5, consistently outperforms the available opponents.

5 Critique of the main results

5.1 CFR critique

In general, CFR introduces a more scalable method for solving extensive-form games with imperfect information. Earlier approaches, such as linear and quadratic programming, can be computationally infeasible for large game trees. CFR reframes the computation of Nash equilibria as a learning problem based on regret minimization. A key strength of this approach is that regret is minimized locally at each information set, which enables further opportunities for parallelization and effective abstraction.

On the other hand, CFR is not a one-size-fits-all solution for extensive games. It does not guarantee convergence to a 2ϵ-Nash equilibrium in general-sum or multi-player settings (without modification), restricting its theoretical guarantees to two-player zero-sum games. This limits its applicability to certain abstractions of real-world strategic interactions. Additionally, CFR assumes perfect recall, which holds in well-controlled experiments but may not be realistic in more complex or memory-limited environments. Finally, CFR strategies are learned entirely through self-play and are fixed post-training. This limits the model to offline deployment with a static final average strategy, making it unsuitable for environments where real-time adaptation to dynamic opponents is required.

5.2 Application Specific Critique

This section critiques how the poker game was abstracted, especially given that the author uses performance on this abstraction as a primary evaluation metric for the CFR algorithm. The main point of critique is the use of bucket abstractions. While the hand strength squared metric does effectively reduce the number of game states and makes the computation tractable, it introduces a loss of precision. Hands with different strategic implications, yet close enough by the metric, may fall into the same bucket and be treated identically. This can lead to suboptimal decision-making and potentially exploitable strategies, especially if the bucketing scheme is not well-calibrated. Furthermore, strategies trained on a particular abstraction may not generalize well to the full game or other variants of poker.

The simulations in the paper were conducted on a 2.4GHz dual-core AMD Opteron 280 processor, which was suitable at the time of publication but is now outdated [1]. CFR can be parallelized at the level of information sets, suggesting that modern multi-core CPUs or GPU clusters could significantly accelerate training. Even without parallel deployment, improvements in CPU performance make it feasible to increase the number of buckets used, thereby allowing abstractions that better reflect the complexity of real poker.

5.3 Historical Context and Novelty

CFR introduced a new way to solve extensive-form games with imperfect information. Before this, most methods used linear or quadratic programming, which required solving the entire game globally. These methods did not scale well for large games. CFR introduced the idea of updating regret locally at each information set. In the past, regret minimization was only used in normal-form games, like the Rock Paper Scissors example discussed earlier. CFR made it work for extensive-form games by introducing counterfactual reach probabilities. This allowed the algorithm to compute regret at each decision point separately, making it much more scalable. The main difficulty in deriving CFR was conceptual. Earlier methods focused on solving games globally, and the idea of applying regret minimization locally to each decision point was not obvious. This conceptual leap likely explains why the result was not discovered earlier. Since its introduction, CFR has become a foundational algorithm in computational game theory and has inspired several state-of-the-art poker agents, including DeepStack and Libratus[3][4].

References

[1] M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione, "Regret Minimization in Games with Incomplete Information," *Neural Information Processing Systems*, vol. 20, pp. 1729–1736, Dec. 2007, doi: 10.7939/r3q23r282.

[2] T. Neller and M. Lanctot, "An Introduction to Counterfactual Regret Minimization 1 Motivation," 2013. Available: http://modelai.gettysburg.edu/2013/cfr/cfr.pdf

[3] M. Moravčík, M. Schmid, N. Burch, V. Lisý, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, and M. Bowling, "DeepStack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, vol. 356, no. 6337, pp. 508–513, 2017, doi: 10.1126/science.aam6960.

[4] N. Brown and T. Sandholm, “Superhuman AI for Multiplayer Poker,” *Communications of the ACM*, vol. 62, no. 11, pp. 92–99, 2019.